Visualization of Graphs

Jean-Daniel Fekete, Inria
http://www.aviz.fr/~fekete
Network data

• networks
  – model relationships between things
    • aka graphs
  – two kinds of items, both can have attributes
    • nodes
    • links

• tree
  – special case
  – no cycles
    • one parent per node
Formal definition

• A graph is:
  • A set of Vertices $V=\{v_i\}$
  • A set of edges $E=\{e_j\}$ with $e=(v_s, v_d) \in V \times V$
  • When the order of the couple in $E$ is meaningful, the graph is **directed**, otherwise, it is **undirected**
  • A graph is a mapping of $V$ into $V$
  • From there, we can define several measures or intrinsic properties on a graph
Network tasks: topology-based and attribute-based

- topology based tasks
  - find paths
  - find (topological) neighbors
  - compare centrality/importance measures
  - identify clusters / communities

- attribute based tasks (similar to table data)
  - find distributions, ...

- combination tasks, incorporating both
  - example: find friends-of-friends who like cats
    - topology: find all adjacent nodes of given node
    - attributes: check if has-pet (node attribute) == cat
Issues

- Graph Layout
- Scalability
- Navigation

Beware of the hairball!
Arrange Networks and Trees

- **Node–Link Diagrams**
  - Connection Marks
  - Networks: ✔️
  - Trees: ✔️

- **Adjacency Matrix**
  - Derived Table
  - Networks: ✔️
  - Trees: ✔️

- **Enclosure**
  - Containment Marks
  - Networks: ✗
  - Trees: ✔️
Node-link diagrams

- nodes: point marks
- links: line marks
  - straight lines or arcs
  - connections between nodes
- intuitive & familiar
  - most common
  - many, many variants

Node–Link Diagrams
Connection Marks

Free

Styled

Fixed

HJ Schultz 2006
Criteria for good node-link layouts

• minimize
  – edge crossings, node overlaps
  – distances between topological neighbor nodes
  – total drawing area
  – edge bends

• maximize
  – angular distance between different edges
  – aspect ratio disparities

• emphasize symmetry
  – similar graph structures should look similar in layout
Criteria conflict

- most criteria NP-hard individually
- many criteria directly conflict with each other

Schulz 2004
Optimization-based layouts

• formulate layout problem as optimization problem
• convert criteria into weighted cost function
  − $F(\text{layout}) = a^*[\text{crossing counts}] + b^*[\text{drawing space used}]+...$
• use known optimization techniques to find layout at minimal cost
  − energy-based physics models
  − force-directed placement
  − spring embedders
Afghanistan Stability / COIN Dynamics

Bummiler, 2010
Linear layout

- THREAD ARCS: An Email Thread Visualization (Bernard Kerr, IBM)

- Arc Diagrams: The Shape of Song (Martin Wattenberg, IBM)

- http://www.bewitched.com/
Idiom: **circular layouts / arc diagrams (node-link)**

- restricted node-link layouts: lay out nodes around circle or along line
- data
  - original: network
  - derived: node ordering attribute (global computation)
- considerations: node ordering crucial to avoid excessive clutter from edge crossings
  - examples: before & after barycentric ordering

[Image of circular layouts]

[Image of barycentric ordering examples]
Linear Layout Theory

• Order vertices so that a structure appears
  • Layout $\varphi$
• Some well-known objective functions (Diaz et al. 02):
  • Bandwidth
  • MinLA
  • Cut Width
  • Min Cut
  • Sum Cut

**Fig. 1.** A graph $G$ together with some layout measures and a graphical representation of the layout $\varphi = \{(a, 1), (b, 5), (c, 3), (d, 7), (e, 8), (f, 6), (g, 4), (j, 9), (h, 2)\}.$
Circular Layout

- Egocentric Social Networks
- Can be ordered too
centrality visualization
Northway (1940)
centrality visualization
network of organizations involved in policy making
Animation of a circular network
[Animated Exploration of Graphs with Radial Layout, InfoVis’01]

https://youtu.be/OPX5iGro_IA
Free Layout

- Order is not meaningful, only proximity is
- Force-directed methods
- Treemaps and Venn Diagrams

for $t \leftarrow 1$ to $\text{ITERATIONS}$ do
  for $v \in V$ do
    $D \leftarrow \sum_{u:\{u,v\} \not\in E} f_{\text{rep}}(p_u, p_v) + \sum_{u:\{u,v\} \in E} f_{\text{spring}}(p_u, p_v)$;
    $p_v \leftarrow p_v + \delta \cdot D$
**Idiom: force-directed placement**

- **visual encoding**
  - link connection marks, node point marks
- **considerations**
  - spatial position: no meaning directly encoded
    - left free to minimize crossings
  - proximity semantics?
    - sometimes meaningful
    - sometimes arbitrary, artifact of layout algorithm
    - tension with length
      - long edges more visually salient than short
- **tasks**
  - explore topology; locate paths, clusters
- **scalability**
  - node/edge density $E < 4N$

Force-directed Layout

- Spring Model: \( f_{\text{rep}} \) linear, \( f_{\text{att}} \) linear
  - Good space filling when the network is sparse and homogeneous
- Kamada&Kawai: \( f_{\text{rep}} \) linear, \( f_{\text{att}} \) function of the Graph-theoretical distance
  - Shows the shortest paths as quasi straight lines
- Fruchterman&Reingold: \( f_{\text{att}} \) quadratic
  - Fills the space better for varying degrees
- Davidson&Harel: \( f_{\text{rep}} \) linear, \( f_{\text{att}} \) linear*degree
  - Better space filling for heterogenous graphs
- Hall/Spectral
  - Very fast but works best with grids
- LinLog: \( f_{\text{rep}} \) linear, \( f_{\text{att}} \) linear+log
  - Show clusters
- New algorithms all the time
- Now, merging with multidimensional projections
Examples

Fruchterman-Reingold model    Node-repulsion LinLog Edge-repulsion LinLog model
Ordered layout

- One dimension is orderable (e.g. genealogical trees)
  - The other can be partially ordered
- Sugiyama algorithm
- Improved by C. North [A Technique for Drawing Directed Graphs.]
3D Networks

- 3D methods usually extend to 3D naturally
- Navigation is then required
Diagrams and Matrices

Directed network

A → B → C → D

Adjacency matrix

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Source

Destination/target
Adjacency matrix representations

- derive adjacency matrix from network
Adjacency matrix examples

HJ Schulz 2007
Node order is crucial: Reordering

https://bost.ocks.org/mike/miserables/
InfoVis Co-authoring (K. Börner et al.)
Generally, after loading...
Explore

Communicate
Structures visible in both

http://www.michaelmcguffin.com/courses/vis/patternsInAdjacencyMatrix.png
Adjacency matrix
Idiom: **adjacency matrix view**

- **data: network**
  - transform into same data/encoding as heatmap

- **derived data: table from network**
  - 1 quant attrib
    - weighted edge between nodes
  - 2 categ attribs: node list x 2

- **visual encoding**
  - cell shows presence/absence of edge

- **scalability**
  - 1K nodes, 1M edges


Node-link vs. matrix comparison

• node-link diagram strengths
  − topology understanding, path tracing
  − intuitive, flexible, no training needed

• adjacency matrix strengths
  − focus on edges rather than nodes
  − layout straightforward (reordering needed)
  − predictability, scalability
  − some topology tasks trainable

• empirical study
  − node-link best for small networks
  − matrix best for large networks
  • if tasks don’t involve path tracing!

Readability Experiment
Controlled Experiment: Node Link Diagrams vs. Adjacency Matrices

The Tasks:
- Tasks related to the overview
  - Number of vertices
  - Number of arcs
- Tasks related to graph elements
  - Finding an element (a vertex, a link)
  - Finding the most connected vertex (a central actor, a pivot, a hub)
  - Finding a common neighbor
  - Finding a path
- Random graphs (3 sizes et 3 densities)
- 2 representations: Node-Link + Matrix

Results:
- Node-link diagrams are preferable for small sparse graphs (20 vertices)
- Matrices are more readable wrt dense graphs and medium/large graphs (>20 vertices)

Completion time for the 7 tasks, 3 densities and 2 representations (Node-Link in blue, Matrix in red)

References:
Matrix vs. NodeLink

- Usable without reordering
- No node overlapping
  No edge crossing
  → Readable for dense graphs
- Fast navigation
- Fast manipulation
  → Usable interactively
- More readable for some tasks
- Less familiar
- Use more space
- Weak for path following tasks

- Familiar
- Compact
- More readable for path following
- More effective for small graphs
- More effective for sparse graphs

- Useless without layout
- Node overlapping
  Edge crossing
  → Not readable for dense graphs
- Manipulation requires layout computation
Idiom: **NodeTrix**

- hybrid nodelink/matrix
- capture strengths of both

More Graph Drawing: Biofabric

Hypergraph?

= a generalization of a graph
where EDGES can connect
one, two, or more VERTICES
Graph (e.g. who knows who)
Node link diagram
Hypergraph (e.g. teams)

Catherine    Jean-Daniel
Catherine    Paolo
Paolo        PaolA    Jean-Daniel
Paolo        PaolA    Jean-Daniel    Catherine
Looks just the same as…
Looks just the same as… the “who knows who” graph
?? How to reveal the hyperedges ??
Multiple links – different link attributes
Teams = Sets
Bi-partite network (nodes and teams)
Hypergraph

Time 1

Catherine

Jean-Daniel

Paolo

PaolA
Dynamic Hypergraph (changes overtime)
Parallel Aggregated Ordered Hypergraph
PAOHvis
Node-Link Representation
Lisibilité des liens ?

- 3 codages
- 3 densités
- Plusieurs graphes aléatoires
- On pose des questions
- On mesure
  - Le temps de réponse
  - La validité des réponses
- On regarde s'il y a une différence statistique
- Oui !


Edge bundling
Edge Bundling ou pas ?

Edge Bundling Methods

(a) No bundling

(b) Metro-Style Bundling

(c) Edge Bundling (complete)

(d) Edge Bundling (relaxed)

(e) Power Graphs

(f) Confluent Drawing

Confluent Graph

https://aviz.fr/~bbach/confluentgraphs/
Edge Bundling ou pas ?

Les algorithmes de groupement des liens simplifient le dessin de réseaux

Mais ils produisent des résultats difficiles à contrôler

- Si l'objectif est de rendre le réseau plus beau, OK
- Si les trajectoires doivent avoir un sens, alors il faut les éviter
- Simplifier retire de l'information...

Trees
Arrange Networks and Trees

- **Node–Link Diagrams**
  - Connection Marks
  - ![Networks and Trees](image)

- **Adjacency Matrix**
  - Derived Table
  - ![Networks and Trees](image)

- **Enclosure**
  - Containment Marks
  - ![Networks and Trees](image)
Node-link trees

- Reingold-Tilford
  - tidy drawings of trees
    - exploit parent/child structure
  - allocate space: compact but without overlap
    - rectilinear and radial variants


- nice algorithm writeup

http://billmill.org/pymag-trees/

http://bl.ocks.org/mbostock/4339184

http://bl.ocks.org/mbostock/4063550
Idiom: **radial node-link tree**

- **data**
  - tree

- **encoding**
  - link connection marks
  - point node marks
  - radial axis orientation
    - angular proximity: siblings
    - distance from center: depth in tree

- **tasks**
  - understanding topology, following paths

- **scalability**
  - 1K - 10K nodes (with/without labels)

Link marks: Connection and containment

• marks as links (vs. nodes)
  - common case in network drawing
  - 1D case: connection
    • ex: all node-link diagrams
    • emphasizes topology, path tracing
    • networks and trees
  - 2D case: containment
    • ex: all treemap variants
    • emphasizes attribute values at leaves (size coding)
    • only trees

Idiom: **treemap**

- **data**
  - tree
  - 1 quant attrib at leaf nodes

- **encoding**
  - area containment marks for hierarchical structure
  - rectilinear orientation
  - size encodes quant attrib

- **tasks**
  - query attribute at leaf nodes
  - ex: disk space usage within filesystem

- **scalability**
  - 1M leaf nodes

---

**Enclosure**

Containment Marks

![SequoiaView](https://www.win.tue.nl/sequoiaview/)

Idiom: implicit tree layouts (sunburst, icicle plot)

- alternative to connection and containment: position
  - show parent-child relationships only through relative positions

<table>
<thead>
<tr>
<th>Treemap</th>
<th>Sunburst</th>
<th>Icicle Plot</th>
</tr>
</thead>
<tbody>
<tr>
<td>containment</td>
<td>position (radial)</td>
<td>position (rectilinear)</td>
</tr>
</tbody>
</table>
Idiom: implicit tree layouts (sunburst, icicle plot)

- alternative to connection and containment: position
  - show parent-child relationships only through relative positions

Treemap
containment
only leaves visible

Sunburst
position (radial)
inner nodes & leaves visible

Icicle Plot
position (rectilinear)
inner nodes & leaves visible
Idiom: implicit tree layouts (sunburst, icicle plot)

- alternative to connection and containment: position
  - show parent-child relationships only through relative positions

Treemap
- containment
- only leaves visible

Sunburst
- position (radial)
- inner nodes & leaves visible

Icicle Plot
- position (rectilinear)
- inner nodes & leaves visible
Tree drawing idioms comparison

Comparison: tree drawing idioms

- data shown
  - link relationships
  - tree depth
  - sibling order
Comparison: tree drawing idioms

- data shown
  - link relationships
  - tree depth
  - sibling order

- design choices
  - connection vs containment link marks
  - rectilinear vs radial layout
  - spatial position channels

Comparison: tree drawing idioms

• data shown
  – link relationships
  – tree depth
  – sibling order

• design choices
  – connection vs containment link marks
  – rectilinear vs radial layout
  – spatial position channels

• considerations
  – redundant? arbitrary?
  – information density?
    • avoid wasting space
    • consider where to fit labels!

[Quantifying the Space-Efficiency of 2D Graphical Representations of Trees.
treevis.net: Many, many options!

https://treevis.net/
Multilevel networks

• derive cluster hierarchy of metanodes on top of original graph nodes

[Schulz 2004]
Idiom: **GrouseFlocks**

- **data:** compound network
  - network
  - cluster hierarchy atop it
    - derived or interactively chosen

- **visual encoding**
  - connection marks for network links
  - containment marks for hierarchy
  - point marks for nodes

- **dynamic interaction**
  - select individual metanodes in hierarchy to expand/contract

Idiom: **sfdp** (multi-level force-directed placement)

- data: compound graph
  - original: network
  - derived: cluster hierarchy atop it

- considerations
  - better algorithm for same encoding technique
    - same: fundamental use of space
    - hierarchy used for algorithm speed/quality but not shown explicitly

- scalability
  - nodes, edges: 1K-10K
  - hairball problem eventually hits

---

Idiom: **hierarchical edge bundling**

- data
  - any layout of compound network
    - network: software classes (nodes), import/export between classes (links)
    - cluster hierarchy: class package structure
  - derived: bundles of edges with same source/destination (multi-level)

- idiom: curve edge routes according to bundles

- task: edge clutter reduction

Hierarchical edge bundling

- works for any layout: treemap vs radial

Free Software

- Java
- C/C++/Python
  - Tulip [https://tulip.labri.fr/site/](https://tulip.labri.fr/site/)
- Web
  - D3 [https://d3js.org/](https://d3js.org/)
  - Vega [https://vega.github.io/vega/examples/](https://vega.github.io/vega/examples/)
  - The Vistorian [https://vistorian.net/](https://vistorian.net/)
  - [https://demo.hedgedoc.org/features](https://demo.hedgedoc.org/features)
  - WebCola [https://ialab.it.monash.edu/webcola/](https://ialab.it.monash.edu/webcola/)